(LISTING W/INTEL)

15 COMMANDS

STU HAIGH
1557 WARBLER WAY
SUNNYVALE CA 94087
(408) 738-4616

```
        TITLE   'ARCADIAN RDOS 1.0 '
;
;       COPYRIGHT (C) 1980, STU HAIGH
;
;  THIS IS A CP/M COMPATABLE RESEDENT DISK OPERATING SYSTEM.
;  THIS CODE IS DESIGNED TO INTERFACE INTO THE CROMEMCO SOFTWARE
; SYSTEM AND IS PROVIDED WITH AN AUTOLOAD FEATURE THAT WILL LOAD
; TRACK ZERO, SECTOR ZERO OF DRIVE A STARTING AT RAM LOACTION 0080.
; CONTROL WILL THEN BE PASSED TO THE JUST LOADED CODE AT LOCATION 0080.
;
;
        ORG     0C000H          ; START OF SULK POWER ON MONITOR
;
;
DSTACK: EQU     0007CH          ; MUST LEAVE ROOM FOR 4 BYTES OF
;                                 TEMPORARY STORAGE ABOVE STACK.
;
;
;               (DSTACK)      = DISK FLAGS
;
;             BIT  ASSIGNMENT  FOR  THE  DISK  FLAGS
;
FASTSEEK: EQU   7               ; 1 = FAST SEEK  0 = SLOW SEEK
DISKMODE: EQU   5               ; 1 = _________  0 = _________
MAXI:     EQU   4               ; 1 = MAXI   0 = MINI
;
; THE DISK NUMBER <0-3> OCCUPIES BITS 0 & 1 OF (DSTACK)
;
;               (DSTACK+1)  =  DISK LETTER (A - D)
;               (DSTACK+2)  =  ROOM FOR UPTO 2 SEMI-COLONS AS PART
;               (DSTACK+3)       OF THE DISK PROMPT.
```

Bob.

this CODE USES A 5501 AS A COM. CTRLer AND A 1771 FLex DISK CTrL. IT WILL SUPPORT FOUR 5¼ INCH, OR 2-5¼ & 1-9, OR 2-9 INCH DISK DRIVES.

I HAVE ALL of The DOCUMENTATION, THAT IS A USERS MANUAL. I CAN LET YOU HAVE COPY, LIMITED, of the CODE BURNED INTO 2716's IF YOU WANT TO TRY IT OUT. I DIDNIT GET AROUND TO COPYING the 'MANUAL' BUT IF YOU WANT IT JUST DROP A LINEPr CALL.

P.S. CAN ALSO REORG AND BURN FOR ANY LOCATION other than ØCØØØh.

```
;
;       FD1771 FLEXIBLE  DISK  DRIVE  INTERFACE  ADDRESSES
;
NDRIVES: EQU      4                ;MAX. NO. OF DISK DRIVES
;
DCOMMND: EQU      30H              ; OUTPUT - COMMAND REG.
DSTAT:   EQU      30H              ; INPUT - STATUS REG.
DTRACK:  EQU      31H              ; TRACK REG.
DSEC:    EQU      32H              ; SECTOR REG.
DDATA:   EQU      33H              ; DATA REG.
DCONTR:  EQU      34H              ; OUTPUT - CONTROL REG.
DFLAGS:  EQU      34H              ; INPUT - FLAGS REG.
;
DAXCMD:  EQU      04H              ; AUXILIARY COMMAND REGISTER
                                   ;   PARALLEL DATA OUTPUT
DAXSTA:  EQU      04H              ; AUXILIARY STATUS REGISTER
                                   ;   PARALLEL DATA INPUT
;
MAXIM:   EQU      10H              ;MASK FOR MAXI DISK
HDLDM:   EQU      20H              ;HEAD LOAD MASK
;
ERRMASK: EQU      98H              ;DISK ERROR MASK
```

```
;
;                       AUTO-IPL  CONTROL
;
BOTDISK: EQU      00H               ; IPL DRIVE (DRIVE A)
BOTSECT: EQU      01H               ; IPL SECTOR (1)
;
BOTBUFF: EQU      80H               ; ADDRESS OF IPL INPUT BUFFER
;
BOTSTRT: EQU      80H               ; START ADDRESS OF IPL'ED PROGRAM
;
BOTSW:   EQU      40H               ; AUTO-IPL ENABLE SWITCH MASK
;                                     <ON = IPL; OFF = GOTO START>
```

```
;
;       TMS 5501  SYSTEM  CONSOLE  I/O INTERFACE  ADDRESSES
;
BAUD:     EQU       00H             ; OUTPUT - UART BAUD RATE REG.
STAT:     EQU       00H             ; INPUT - UART STATUS REG.
DATA:     EQU       01H             ; I/O - UART DATA REG.
COMMND:   EQU       02H             ; OUTPUT - UART COMMAND REG.
IADDR:    EQU       03H             ; INPUT - INTERRUPT ADDRESS REG.
IMASK:    EQU       03H             ; OUTPUT - INTERRUPT MASK REG.
INPAREL:  EQU       04H             ; INPUT - PARALLEL DATA REG.
                                    ;  AUXILIARY DISK STATUS REG.
OTPAREL:  EQU       04H             ; OUTPUT - PARALLEL DATA REG.
                                    ;  AUXILIARY DISK COMMAND REG.
TIMEA:    EQU       05H             ; OUTPUT - TIMER ONE COUNT REG.
TIMEB:    EQU       06H             ; OUTPUT - TIMER TWO COUNT REG.
TIMEC:    EQU       07H             ; OUTPUT - TIMER THREE COUNT REG.
TIMED:    EQU       08H             ; OUTPUT - TIMER FOUR COUNT REG.
TIMEF:    EQU       09H             ; OUTPUT - TIMER FOUR COUNT REG.
;
DAV:      EQU       40H             ; DATA AVAILABLE MASK
TBE:      EQU       80H             ; XMITTER BUFFER EMPTY MASK
;
;           SYSTEM CONSOLE  ASCII  CHARACTER  CODE
;
CR:       EQU       0DH
LF:       EQU       0AH
ESC:      EQU       1BH
ALT:      EQU       7DH
;
CASE:     EQU       00H             ; DATA CASE MASK
;
```

ENTRY POINT ON PWR ON or Reset

```
;
;- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
;
;                RDOS 1.0 POWER ON MONITOR ENTRY POINT
;
;  SET THE POWER ON 'JUMP START' TO THE ADDRESS OF 'START:' <0C00>.
;
;- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
;
START:                                  ; <ENTRY POINT>
        NOP                             ;WAIT FOR
        NOP                             ;SYSTEM TO
        NOP                             ;SETTLE DOWN
        DI                              ;DISABLE INTERRUPTS
        LD      A,00H                   ; CLEAR THE 'A' REG.
        LD      I,A                     ; INITIALIZE I REG
        LD      R,A                     ;INITIALIZE R REG
        IM      0                       ;SET INTERRUPT MODE ZERO.
        LD      IX,DSTACK+4             ; SET IX TO INIT SP VALUE
        LD      HL,DSTACK
        LD      SP,HL
        EX      DE,HL                   ; DE -> TEMP STORAGE
        CALL    INITBAUD                ; INITIALIZE SYSTEM CONSOLE PORT
        LD      A,00H                   ; CLEAR A REG.
        OUT     IMASK,A                 ; INHIBIT SYSTEM CONSOLE INTERRUPTS
        IN      A,DFLAGS                ; READ DISK FLAGS-BOOT SWITCH
        AND     BOTSW                   ; TEST AUTO IPL SWITCH
  ;                                       ON = LOAD CDOS. OFF=EXEC RDOS
        JP      Z,BOOTDK                ; IPL CDOS
        JP      MONITR
```

```
;
;     * * * MONITOR COMMAND * * *
;
;      --QUIT RDOS & IPI CDOS--
;
; FUNCTION: IPL TRACK ZERO, SECTOR ZERO FROM DRIVE A.
;
; FORMAT:  'B (RETURN)'
;
;
BOOTMC:
          CALL      PMSGFOLLOWING
          DB        CR,':-VERIFY THAT THE CDOS DISK IS MOUNTED ON'
          DB        ' DRIVE A.'
          DB        CR,':-PRESS <RETURN> WHEN READY','.'+80H
          CALL      SKSGCR              ;REQUIRE A CR
;
BOOTDK:
          LD        A,DAV               ; RESET DISK DRIVE CONTROLLER
          OUT       DCOMMND,A
BOT200:
          IN        A,DSTAT             ;READ DISK STATUS
          RRA
          JR        C,BOT200
          DI
          LD        A,1.SHL.MAXI        ;MAXI FLAG
BOT300:
          LD        HL,BOTBUFF          ;INIT. BUFFER PNTR
          LD        SP,HL               ;& STACK PNTR
          PUSH      AF                  ;SAVE MINI/MAXI FLAG
          LD        B,BOTDISK           ;GET BOOT DISK ADDRESS
          CALL      DHOME               ;HOME DISK
          JR        NZ,BOT500           ; DISK ERROR
          POP       AF                  ;GET MINI/MAXI FLAG
          PUSH      AF
;
;  THIS DEFINES THE BOOT DISK ATTRIBUTES
;
          LD        B,BOTDISK           ;GET BOOT DISK ADDRESS
          LD        E,BOTSECT           ; GET BOOT SECTOR
          CALL      DREAD               ;READ THE SECTOR
          JP        Z,BOTBUFF           ;OK, GO EXECUTE
BOT500:
          POP       AF                  ;GET MINI/MAXI FLAG
          XOR       1.SHL.MAXI          ;TOGGLE IT
          JR        BOT300
;
;
; HOME DISK DRIVE
;
; INPUT - B CONTAINS DISK NUMBER (0,1,2,3)
;         A BIT 4 CONTAINS 1 IF MAXI
;
; OUTPUT - B CONTAINS STATUS
;         ZERO FLAG RESET IF ERROR
;
```

```
; REGISTERS A,F,B,C ARE CHANGED
;
;
DHOME:                                  ;HOME THE DISK DRIVE
        CALL    SELECT                  ;SELECT DISK
        OUT     DCONTR,A                ;OUTPUT SELECT BYTE
        LD      D,ERRMASK               ;ERROR MASK
        AND     MAXIM                   ;MAXI DISK ?
        LD      A,7FH                   ;TURN OFF HIGH SPEED SEEK
        OUT     OTPAREL,A
        LD      A,0FH                   ;LOAD MINI RESTORE COMMAND
        JR      Z,EXECUTE               ;NO, ITS A MINI
        LD      A,0DH                   ;MAXI RESTORE COMMAND
        JR      EXECUTE                 ;EXEC COMMAND &
;                                       ;WAIT TILL DONE
;
;
; SEEK TO DESIRED TRACK
;
; TRACK REGISTER MUST HAVE BEEN PREVIOUSLY LOADED
; (MAY BE DONE BY INITIALLY DOING A HOME)
;
; INPUT - B CONTAINS DISK DRIVE (0,1,2,3)
;         D CONTAINS TRACK
;         A BIT 7 = 1 FOR FAST SEEK
;         A BIT 4 = 1 FOR MAXI DISK / 0 FOR MINI DISK
;
; OUTPUT - B CONTAINS STATUS
;                                       ZERO FLAG RESET IF ERROR
;
; REGISTERS A,F,B,C,D ARE CHANGED
;
DSEEK:
        PUSH    AF                      ;SAVE DISK FLAG
        CALL    SELECT                  ;SELECT DISK
        OUT     DCONTR,A                ;OUTPUT CONTROL BYTE
        OUT     (C),D                   ;OUTPUT DESIRED TRACK
        LD      D,ERRMASK               ;ERROR MASK
        POP     AF                      ;GET FLAGS
        RLA                             ;FAST SEEK?
        JR      C,DSK500
        AND     MAXIM.SHL.1             ;MASK FOR MINI/MAXI
        LD      A,01FH                  ;LOAD SEEK COMMAND FOR MINI
        JR      Z,EXECUTE               ;MINI DISK
        LD      A,1DH                   ;LOAD COMMAND FOR MAXI
;
;
EXECUTE:
        OUT     DCOMMND,A               ;OUTPUT COMMAND
;
EXCCHK:
        IN      A,DFLAGS                ;WAIT FOR COMPLETION
        RRA
        JR      NC,EXCCHK               ;(UNTIL INTREQ)
EREXIT:
        IN      A,DSTAT                 ;DISK STATUS
```

```
          LD        B,A             ;SAVE STATUS
          AND       D               ;MASK FOR ERRORS
          RET
;
;
DSK500:
          LD        A,6FH           ;TURN ON FAST SEEK
          OUT       OTPAREL,A
          LD        A,18H           ;SEEK COMMAND
          CALL      EXECUTE
DSK540:
          IN        A,INPAREL       ; FAST SEEK DONE?
          AND       40H
          JR        NZ,DSK540
          LD        A,7FH           ;TURN OFF FAST SEEK
          OUT       OTPAREL,A
          SUB       A               ;NO ERROR CHECKING, SAY OK
          LD        B,A
          RET
;
;
;          * * READ 1 SECTOR FROM DISK * *
;
; INPUT - B CONTAINS DISK (0,1,2,3)
;         E CONTAINS SECTOR
;         A BIT 4 = 1 FOR MAXI & 0 FOR MINI DISK
;         HL CONTAINS BUFFER ADDRESS
;
; OUTPUT - B CONTAINS STATUS
;          Z FLAG IS SET IF NO ERRORS
;          HL PTS PAST BUFFER
;
; REGISTERS  A,F,B,C,E,H,L  ARE CHANGED
;
;
DREAD:
          CALL      SETUP           ;SET UP FOR READ
          ADD       A,88H           ;ADD READ COMMAND TO
                                    ;HEAD LOAD FLAG
          LD        D,9CH           ;ERROR MASK
;
          OUT       DCOMMND,A       ;OUTPUT READ COMMAND
DRD250:
          IN        A,DFLAGS        ;WAIT FOR REQUEST
          RRA                       ;CHECK FOR INTREQ
          JR        C,EREXIT        ;END OF SECTOR OR ERROR
          INI                       ;READ A BYTE
          JP        NZ,DRD250       ;NOT DONE YET
          JR        EXCCHK          ;WAIT FOR INTREQ
;
;  WRITE A SECTOR TO THE DISK
;
; INPUT - B CONTAINS DISK (0,1,2,3)
;         E CONTAINS SECTOR
;         A BIT 4 = 1 FOR MAXI  0 FOR MINI DISK
;         HL CONTAINS BUFFER ADDRESS
```

```
;
; OUTPUT - B CONTAINS STATUS
;          Z FLAG IS SET IF NO ERRORS
;          HL PTS PAST BUFFER
;
; REGISTERS  A,F,B,C,D,E,L  ARE CHANGED
;
DWRITE:  CALL     SETUP            ;SET UP FOR WRITE
         ADD      A,0A8H           ;ADD WRITE COMMAND TO
                                   ;HEAD LOAD FLAG
         LD       D,0FCH           ;ERROR MASK
         OUT      DCOMMND,A        ;OUTPUT WRITE COMMAND
DWR250:
         IN       A,DFLAGS         ;WAIT FOR REQUEST
         RRA                       ;CHECK FOR INTREQ
         JR       C,ERFXIT         ;END OF SECTOR OR ERROR
         OUTI                      ;READ A BYTE
         JP       NZ,DWR250        ;NOT DONE YET
         JR       EXCCHK           ;WAIT FOR INTREQ
;
;
;   SET UP FOR READ OR WRITE
;
; INPUT -  B CONTAINS DISK DRIVE (0,1,2,3)
;          E CONTAINS SECTOR
;          A BIT 4 CONTAINS  1 IF MAXI & 0 FOR MINI DISK
;
; OUTPUT - D CONTAINS SELECT BYTE
;          A CONTAINS HEAD LOAD FLAG
;          B CONTAINS 128 (# OF BYTES)
;          C CONTAINS DATA PORT
;
; REGISTERS A,F,B,C,D ARE CHANGED
;
SETUP:
         CALL     SELECT           ;GET SELECT BYTE
         OR       80H              ;TURN ON AUTO WAIT
         LD       D,A              ;SAVE CONTROL BYTE
         LD       A,E              ;SECTOR #
         OUT      DSEC,A
;
; CHECK TO SEE IF THE DISK HEAD IS LOADED
;
         IN       A,DFLAGS         ;READ FLAGS
         AND      HDLDM            ;HEAD LOADED?
         LD       A,D              ;CONTROL BYTE
         OUT      DCONTR,A         ;(THIS MUST BE DONE AFTER
                                   ;THE INPUT FROM DFLAGS
                                   ;BECAUSE OF AUTO WAIT)
         LD       A,4              ;HEAD NOT LOADED
         RET      Z
         SUB      A                ;HEAD LOADED
         RET
;
;
; SELECT DISK DRIVE
```

```
;
; INPUT -  B CONTAINS DISK DRIVE (0,1,2,3)
;          A BIT 4 CONTAINS 1 IF MAXI & 0 FOR MINI DISK
;
; OUTPUT - A CONTAINS SELECT BYTE
;          B CONTAINS 128
;          C CONTAINS DATA PORT #
;
; REGISTERS  A,F,B,C  ARE CHANGED
;
SELECT:
         AND      MAXIM              ;GET MAXI FLAG ONLY
         LD       C,A                ;SAVE FLAG
         INC      B                  ;CALCULATE DISK SELECT
         SUB      A
         SCF
SEL300:
         RLA
         DJNZ     SEL300
         OR       C                  ;MAXI FLAG
         OR       20H                ;MOTOR ON
         LD       BC,80C0H+DDATA
         RET
;
;
;
; CHECK INPUT & RETURN WITH DATA IF READY
;
;   TEST SYSTEM CONSOLE INTERFACE IF DATA AVAILABLE LOAD DATA BYTE
;   INTO A-REG AND RETURN.
;   IF NOT DATA AVAILABLE THEN ZERO A-REG AND RETURN.
;
CHKIN:
         IN       A,STAT             ; GET 5501 STATUS REGISTER
         AND      DAV                ; TEST IF 'RECEIVER DATA AVAILABLE'
         RET      Z                  ; RETURN IF NO DATA RECEIVED
         IN       A,DATA             ; GET RECEIVED CHARACTER.
         RET
;
;
; GET CHARACTER FROM INPUT
;
GBYTE:
         CALL     CHKIN              ; GET A CHARACTER
         JR       Z,GBYTE            ; IF CHARACTER = ZERO REPEAT
         AND      7FH                ; SET PARITY BIT TO ZERO.
         RET
```

```
;
;       * * * MONITOR COMMAND * * *
;
; FUNCTION: INITIALIZE BAUD RATE OF THE SYSTEM CONSOLE PORT.
;
; FORMAT: 'I (RETURN)'
;
;
;                AUTO BAUD RATE TABLE
;
;  NOTE: AUTO BAUD RATES ARE BASED ON A 1 MHZ CLOCK RATE.
;
BAUDRS:  DB      090H            ; 19,200B; 1 STOP BIT
         DB      0C0H            ;  9,600B; 1 STOP BIT
         DB      0A0H            ;  4,800B; 1 STOP BIT
         DB      090H            ;  2,400B; 1 STOP BIT
         DB      088H            ;  1,200B; 1 STOP BIT
         DB      084H            ;    300B; 1 STOP BIT
         DB      082H            ;    150B; 1 STOP BIT
         DB      001H            ;    110B; 2 STOP BIT
;
;
INITBR:
         CALL    SKSGCR          ;REQIRE A CR
;
; PUSH CARRIAGE-RETURN TO SELECT THE PROPER BAUD
; RATE FOR THE CURRENT TERMINAL. (THE MAXIMUM
; NUMBER OF CARRIAGE-RETURNS REQUIRED IS FOUR.)
;
; THE FOLLOWING WILL ALSO BE ESTABLISHED:
;
;   A) SYSTEM CONSOLE INTERRUPTS ENABLED,
;   B) RST7 SELECT - OFF, 'DRQ' WILL NOT CAUSE AN INTERRUPT.
;   C) TRANSMITTER BREAK OFF
;
INITBAUD:
         LD      HL,BAUDRS       ; LOAD BAUD RATE TABLE POINTER
         LD      C,BAUD          ; SET BAUD RATE REGISTER
         LD      A,19H           ; INITIALIZE 5501- TEST MODE OFF,
                                 ;   HIGH BAUD RATE, INTER. ENA ON,
                                 ;   RST7 OFF, BREAK OFF
IT1:
         OUT     COMMND,A        ; OUTPUT 5501 COMMAND.
         OUTI                    ; SEND BAUD RATE SELECTION
         CALL    GBYTE           ; GET FIRST BYTE
         CALL    GBYTE           ; GET SECOND BYTE
         CP      CR              ; TEST FOR A CARRAGE RETURN
         LD      A,9             ; REINITIALIZE 5501 - TEST MODE OFF,
                                 ;   LOW BAUD RATE, INTER. ENA ON,
                                 ;   RST7 OFF, BREAK OFF
         JR      NZ,IT1          ; IF NOT (CR) REPEAT
         RET
```

```
;
;       * * * MONITOR COMMAND * * *
;
; FUNCTION: CHANGE THE LOCATION OF THE SYSTEM STACK
;
; FORMAT: 'K <NEW-STACK-LOCATION> (RETURN)'
;
KICKSTK:
        CALL    L1NCR
        JR      LOADIX          ;IX STORES INITIAL SP VALUE
;
;-------------------------------------------------------
;                       MONITOR ENTRY POINT
;-------------------------------------------------------
;
; ENTER MONITOR WITH THE STK PNTR LOADED & WITH
; DE -> THE DISK FLAGS. (THIS IS ALSO
; THE TOP IF THE STACK.)
;
MONITR:
        CALL    PMSGFOLLOWING
        DB      CR,' ARCADIAN  RDOS  1.0  ACTIVE'
        DB      CR+80H
;
LOADIX:
        SUB     A
        LD      (DE),A          ;CLEAR DISK MODE
        PUSH    DE
        POP     IX              ;IX STORES INITIAL SP VALUE
;
CLEANSTACK:
        LD      SP,IX           ;RE-INITIALIZE SP
;
;
; GET COMMAND.
; RETURNS VALUE IN HL & JUMPS TO THAT ADDR.
;
        CALL    CRLF
CMND:
        LD      HL,CMND         ;SET-UP RETURN
        PUSH    IX
        EX      (SP),HL         ;RETN ADDR ON STK
        LD      C,(HL)          ; HL -> DISK FLAGS
        BIT     DISKMODE,C
        INC     HL              ; -> DISK LETTER
        CALL    NZ,PMSG         ;DISK MODE PROMPT
        CALL    PMSGFOLLOWING
PROMPT: DB      ';'+80H         ;THE REGULAR PROMPT
;
        CALL    SKSGC           ;GET THE COMMAND
        JR      NZ,CM6
        LD      (IX),0          ;CR. RESET DISK MODE.
        RET
;
CM6:
        SUB     'A'+CASE        ; < 'A' ?
```

```
        JP      C,INVCMD
        CP      'Z'-'A'+1       ; > 'Z' ?
        JP      NC,INVCMD
        LD      E,A
        LD      D,0
;
        CALL    SKSGO           ;NEXT COMMAND CHARACTER
        CP      ';'
        JP      Z,DISKERR
        EX      DE,HL
        ADD     HL,HL           ;TIMES 2
        LD      DE,CMNDTBL
        ADD     HL,DE           ; + TBL ADDR
        LD      E,(HL)
        INC     HL
        LD      D,(HL)
        EX      DE,HL
        CP      'M'+CASE        ;(USED IN SUBST & DISPL)
        JP      (HL)
;
;
; DISK SELECT
; ENTER WITH E CONTAINING THE DISK NUMBER
;
DISKSELECT:
        LD      A,E             ;DISK NUMBER
        CP      NDRIVES         ;A THROUGH D ONLY
        JP      NC,ERRSEL       ;INVALID DRIVE NUMBER
        LD      B,E             ;SAVE DISK #
        PUSH    IX
        POP     HL              ; -> DISK FLAGS
        OR      1.SHL.DISKMODE+(1.SHL.MAXI)+(1.SHL.FASTSEEK)
        LD      (HL),A          ;DISK # & FLAGS
        LD      D,H
        LD      E,L
        INC     DE              ; -> DISK LETTER
        LD      A,B
        ADD     A,'A'
        LD      (DE),A          ;DISK LETTER
        CALL    GCHR
        CP      ';'
        JR      NZ,DS2
        RES     FASTSEEK,(HL)   ;NOT FAST SEEK
        INC     DE
        LD      (DE),A          ;PART OF DISKMODE PROMPT
        CALL    GCHR
        CP      ';'
        JR      NZ,DS2
        RES     MAXI,(HL)       ;MINI FLOPPY
        INC     DE
        LD      (DE),A
        SUB     A
;
DS2:
        CALL    SKSGCB          ;ALSO EXCGS DE & HL
        SET     7,(HL)          ;MASK END-OF-MSG
```

```
;
        LD      A,(DE)          ;DISK FLAGS
        CALL    DHOME
        LD      A,'H'           ; IN CASE OF HOME ERROR
;
DERRCK:
        RET     Z               ;IF NO ERROR, DONE
;
;
PERRMSG:
        CALL    PMSGFOLLOWING
        DB      CR,':-RDOS DETECTED ERROR.'
        DB      CR,':-ERROR STATUS CODE --->',' '+80H
        CALL    PCHR            ; ERROR LETTER
        LD      A,B             ; ERROR NUMBER
;
;
; PRINT THE 2 HEX DIGITS IN THE A-REGISTER
; AND CLEAN STACK.
;
P2HXCLEAN:
        CALL    P2HEX
        JR      CLEANV
;
;
; PRINT CRLF
;
CRLF:
        LD      A,CR
        JR      PCHR
```

```
;
;       * * * MONITOR COMMAND * * *
;
; FUNCTION: EXAMINE INPUT PORT.
;
; FORTMAT: 'E <PORT NUMBER> (RETURN)'
;
EXMINPUT:
        CALL    L1NCR
        LD      C,E             ;PORT #
        IN      A,(C)
        JR      P2HXCLEAN       ;PRINT THE VALUE, CRLF
;
;
; DISK SELECTION ERROR. RETURNS TO CMD WITH SP
; RE-INITIALIZED.
;
DISKERR:
        CALL    PMSGFOLLOWING
        DB      CR,':-DISK SELECTION ERROR','?'+80H
ESCAPE:
CLEANV:
        JP      CLEANSTACK
;
; GET NEXT SECTOR FOR THE READ & WRITE DISK
; ROUTINES. PRESERVES HL AND, BEFORE RETURNING,
; POPS DE AND BC FROM THE STACK.
;
NEXTSC:
        EXX
        POP     HL              ;RETURN ADDR
        EXX
        POP     DE
        JR      Z,NS2           ;SKIP IF NO ERROR
        DEC     D               ;TRY AGAIN ?
        JR      Z,PERRMSG
        JR      NS4             ; YES. USE OLD MEM PNTR
;
NS2:
        LD      BC,-81H         ; NO ERROR
        ADD     IY,BC           ; BUMP THE INCREMENT
        INC     IY
        EX      (SP),HL         ; USE LATEST MEM PNTR
        LD      D,10            ;RELOAD RETRIAL COUNTER
;
NS4:
        POP     HL              ;MEM PNTR
        POP     BC
        LD      A,C             ;RELOAD DISK FLAGS
        EXX
        PUSH    HL              ;RETURN ADDR
        EXX
        RET     NZ              ;IF ERROR, DONE
;
        CALL    NC,PTRKSC       ;IF NEGATIVE, DONE:
        JR      NC,CLEANV       ;PRINT TRK, SEC, CLEAN STK
```

```
;
        INC     E               ;BUMP SECTOR #
        CALL    CHKSECNO
        RET     NC              ;DONE IF # OK
        IN      A,DTRACK        ;GET TRACK #
        INC     A               ;BUMP IT
        LD      E,A
        PUSH    BC
        CALL    SEEKNXT         ;SEEK NEXT TRACK
        POP     BC
        LD      A,C             ;DISK FLAGS
        LD      E,1             ;SECTOR 1
        RET
;
;
; PRINT SPACE. ALTERS A.
;
SPACE:
        LD      A,' '           ;(CONTINUE BELOW)
;
;
; PRINT THE CHARACTER IN THE A-REGISTER.
; (CHKS INPUT FOR ESC.) PRESERVES ALL REGS.
;
PCHR:
        PUSH    AF              ; SAVE DATA AND FLAGS.
PC1:
        CALL    CHKIN           ; CHECK FOR INPUT
        AND     7FH             ; MASK PARITY BIT
        CP      ESC             ; INPUT = ESC CHAR ?
        JR      Z,ESCAPE        ; IF YES GOTO ESCAPE
        CP      ALT             ; INPUT = ALT MODE ?
        JR      Z,ESCAPE        ; IF YES GOTO ALTER
;
PC2:
        IN      A,STAT          ; GET TRANSMITTER STATUS
        AND     TBE             ; TRANSMIT BUFFER EMPTY ?
        JR      Z,PC2           ; NO, WAIT
        POP     AF              ; RESTORE DATA AND FLAGS
        PUSH    AF
        AND     7FH             ; ZERO PARITY BIT
        OUT     DATA,A          ; TRANSMIT THE CHARACTER
        CP      CR              ; CHAR = <CR> ?
        JR      NZ,PC3          ; NO RETURN
        CALL    PMSGFOLLOWING   ; YES, SEND <LF>
        DB      LF,0CH,80H      ; <LF>,<NUL>,<NUL>+STOP
PC3:
        POP     AF
        RET
;
;
; GET CHARACTER. RETURNS IT IN A.
; ALTERS F.
;
GCHR:
        CALL    GBYTE
```

```
        CALL    PCHR
        CP      61H             ;CONVERT LOWER CASE
        RET     C               ;TO UPPER.
        SUB     20H
        RET
;
;
; LOADS HL WITH SOURCE ADDR, BC & DE
; WITH THE INCREMENT.  ENDS WITH CRLF.
;
L2NCRO:
        SUB     A
;
L2NCR:
        CALL    LD2N
;
; SKIP INITIAL SPACES.
; IF DELIMITER NOT A CR, ERROR
;
SKSGCR:
        CALL    SKSG            ;WAIT FOR NON-SPACE
        JP      NZ,CRERR        ;IF NOT CR, ERROR
        EX      DE,HL
        RET
;
;
; PRINT THE NUMBER IN HL, FOLLOWED BY A COLON.
; PRESERVES ALL REGISTERS EXECPT A.
;
PCADDR:
        CALL    CRLF
;
PADDR:
        CALL    PNHL
        LD      A,':'
        JR      PCHR
```

```
;
;       * * * MONITOR COMMAND * * *
;
; FUNCTION: VERIFY BLOCKS OF MEMORY DATA.
;
; FORMAT: 'V <SOURCE-ADDR> <SOURCE-END> <DESTINATION-ADDR> (RETURN)'
;                           OR
;         'V <SOURCE-ADDR> S <SWATH-WIDTH> <DESTINATION-ADDR> (RETURN)'
;
VERIF:
        CALL    L3NCR           ;GET 3 OPERANDS
;
; COMPARES TWO AREAS OF MEMORY. ENTER WITH
; SOURCE IN HL, DESTINATION IN DE & COUNT
; IN BC. ALTERS ALL REGISTERS.
;
VRFY:
        LD      A,(DE)
        CPI
        DEC     HL
        CALL    NZ,PNHL         ;PRINT SOURCE ADDR
        CALL    NZ,PSNM         ; & CONTENTS
        EX      DE,HL
        CALL    NZ,PSNM         ; & DEST CONTENTS
        CALL    NZ,PSNHL        ; & DEST ADDR
        CALL    NZ,CRLF
        EX      DE,HL
        INC     HL
        INC     DE
        RET     PO              ; IF BC=0, DONE.
        JR      VRFY
```

```
;
;       * * * MONITOR COMMAND * * *
;
; FUNCTION: MOVE A BLOCK OF DATA.
;
; FORMAT: 'M <SOURCE-ADDR> <SOURCE-END> <DESTINATION-ADDR> (RETURN)'
;                                       OR
;         'M <SOURCE-ADDR> S <SWATH-WIDTH> <DESTINATION-ADDR> (RETURN)'
;
MOVE:
        CALL    L3NCR           ;OPERANDS
        PUSH    HL
        PUSH    DE
        PUSH    BC
        LDIR
        POP     BC
        POP     DE
        POP     HL
        JR      VRFY
;
;
;
; LOAD TWO NUMBERS. LOADS DE WITH THE BEGINNING
; ADDR, N1. LOADS BC & HL WITH THE INCREMENT
; N2-N1+1 (OR WITH N2 IF THE OPR IS 'S').
; RETURNS WITH LAST DELIMITER IN A.
;
;
LD2N:
        CALL    GNHL            ;N1 TO HL, DELIM TO A
        EX      DE,HL           ;SAVE N1 IN DE
        CALL    SKSG            ;GET NEXT NON-SPACE
        CP      'S'+CASE        ;SWATH ?
        JR      NZ,L2N1
;
        CALL    GNHL0           ; YES. INCREMENT TO HL
        JR      L2N2
;
L2N1:
        CALL    GNHL            ;INCREMENT
        OR      A               ;CLEAR CY
        SBC     HL,DE           ;N2-N1
        INC     HL              ;INCLUDE END POINT
L2N2:
        LD      B,H
        LD      C,L             ;BC GETS THE INCRM
        PUSH    HL
        POP     IY              ;& SO DOES IY.
        RET
;
;
; LOAD 3 OPERANDS. HL GETS THE SOURCE, BC
; THE INCREMENT, AND DE THE 3RD OPERAND.
;
L3NCR0:
        SUB     A
```

```
;
L3NCR:
        CALL    LD2N
; (CONTINUE BELOW)
;
; ENTER WITH SPACE OR THE FIRST DIGIT
; OF THE NUMBER IN A. LOADS HL WITH
; A NEW NUMBER & THEN EXCHANGES
; DE & HL. FINISHES WITH A CRLF
;
L1NCR:
        CALL    GNHL            ;SKIP SPACES, LOAD HL
        JR      SKSGCR          ;WAIT FOR A CR
;
; CLEARS HL. IF ENTERED WITH HEX CHAR IN A,
; SHIFTS IT INTO HL. O/W, IGNORES LEADING
; SPACES. FIRST CHAR MUST BE HEX. CONTINUES
; SHIFT UNTIL A NON-HEX CHAR RECEIVED & THEN
; RETURNS WITH THE LATTER IN A.
;
; PRESERVES B,C,D,E.
;
;
GNHL0:
        SUB     A
;
GNHL:
        PUSH    BC              ;SAVE
        LD      HL,0            ;CLEAR BUFFER
; STRIP LEADING SPACES & GET CHAR
        CALL    SKSG
; FIRST CHAR MUST BE HEX
        CALL    HEXSH           ;IF HEX, SHIFT INTO HL
        JP      C,HEXERR        ;O/W, ERROR
GN1:
        CALL    GCHR
        CALL    HEXSH           ;IF HEX SGIFT INTO HL
        LD      A,B             ; RESTORE CHAR
        JR      NC,GN1          ; IF HEX, CONTINUE
        POP     BC              ;IF NON-HEX, DONE
        RET
;
;
; IF A CONTAINS HEX CHAR, SHIFTS BINARY EQUIVALENT
; INTO HL. IF NOT HEX, RET WITH CY SET. SAVES
; ORIGINSAL CHAR IN B.
;
HEXSH:
        LD      B,A
        SUB     '0'             ; < '0' ?
        RET     C
        ADD     A,'0'-'G'
        RET     C
        SUB     'A'-'G'
        JR      NC,HX1          ; OK IF >= 'A'
        ADD     A,'A'+CASE-('9'+1)
```

```
        RET     C
HX1:
        ADD     A,'9'+1-'0'
;
; THE A-REG NOW CONTAINS THE HEX DIGIT IN BINARY.
; (THE HIGH-ORDER NIBBLE OF A IS 0.)
;
HXSH4:
        ADD     HL,HL           ; SHIFT 4 BITS INTO HL
        ADD     HL,HL
        ADD     HL,HL
        ADD     HL,HL
        OR      L
        LD      L,A
        RET
;
;
; RETURNS WITH A NON-SPACE IN THE A-REG.
; IF ENTERED WITH A-REG CONTAINING A NULL
; OR A SPACE, GETS NEW CHARS UNTIL FIRST
; NON-SPACE OCCURS, ALTERS AF.
;
SKSGO:
        SUB     A
;
SKSG:
        OR      A               ;DOES A CONTAIN NULL ?
SK1:
        CALL    Z,GCHR
        CP      20H             ;SPACE ?
        JR      Z,SK1
        CP      CR
        RET
;
;
;
; PRINT SPACE FOLLOWED BY THE NUMBER POINTED
; TO BY HL. ALTERS A ONLY.
;
PSNM:
        CALL    SPACE
; (CONTINUE BELOW)
;
; PRINTS THE NUMBER POINTED TO BY HL.
; PRESERVES ALL REGISTERS BUT A.
;
PMN:
        LD      A,(HL)
        JR      P2HEX
;
;
;
; PRINT THE NUMBER IN HL.
; PRESERVES ALL BUT A.
;
PSNHL:
```

```
        CALL    SPACE
;
PNHL:
        LD      A,H
        CALL    P2HEX
        LD      A,L
;                               ;(CONTINUE BELOW)
;
; PRINT THE NUMBER IN THE A-REGISTER.
; PRESERVES ALL REGISTERS.
;
P2HEX:
        CALL    P1HEX
        RRA
P1HEX:
        RRA
        RRA
        RRA
        RRA
        PUSH    AF
        AND     0FH             ;MASK
        CP      10D             ;<= 9 ?
        JR      C,PH1
        ADD     A,7             ;A THRU F
PH1:
        ADD     A,30F           ;ASCII BIAS
        CALL    PCHR            ;PRINT IT
        POP     AF
        RET
;
;
; PRINT MESSAGE. ENTER WITH ADDR OF MSG
; IN HL. THE MESSAGE IS TERMINATED
; AFTER PRINTING A CHARACTER WHOSE
; PARITY BIT WAS SET.
; PRESERVES FLAGS, INCREMENTS HL.
;
PMSG:
        PUSH    AF              ;SAVE
PS1:
        LD      A,(HL)
        INC     HL
        CALL    PCHR
        RLA                     ;LAST CHARACTER ?
        JR      NC,PS1          ;IF NOT, LOOP
        POP     AF
        RET
;
;
; PRINTS THE MESSAGE FOLLOWING THE CALL
; TO THIS ROUTINE.
; PRESERVES ALL REGISTERS
;
PMSGFOLLOWING:
        EX      (SP),HL
        CALL    PMSG
```

```
        EX      (SP),HL
        RET
```

```
;
;       * * * MONITOR COMMAND * * *
;
; FUNCTION: TRANSFER CONTROL TO USER PROGRAM.
;
; FORMAT: 'G <ADDR> (RETURN)'
;
;  EXECUTION WILL RESUME AT LOCATION POINTED TO BY <ADDR>.
;
GO:
        POP     HL              ;CLEAN STACK
        CALL    L1NCR           ;GET ADDR
        EX      DE,HI
        JP      (HL)
```

```
;
;                    * * * MONITOR COMMAND * * *
;
; FUNCTION: DISPLAY MEMORY.
;
; FORMAT: 'DM <STRING ADDR> <ENDING ADDR OR SWATH>'
;
DSPM:
          JP        NZ,INVCMD           ;IF NOT 'M', ERROR
          CALL      L2NCFC              ; GET OPERANDS
DSPM1:
          LD        D,16                ;BYTE COUNT
          CALL      PCADDR              ;ADDRESS
DM2:
          CALL      PSNM                ;MEM CONTENTS
          CPI                           ;INC HL & DEC BC
          JP        PO,CRLF
          DEC       D
          JR        Z,DSPM1
          LD        A,D
          AND       3
          CALL      Z,SPACE
          JR        DM2
;
;
INVCMDB:
          JP        INVCMD              ;ERROR LINKAGE
```

```
;
;       * * * MONITOR COMMAND * * *
;
; FUNCTION: SUBSTITUTE MEMORY OR SEEK TRACK.
;
; FORMAT: 'SM <ADDRESS> (RETURN)'       - SUBSTITUTE MEMORY.
;
; FORMAT: 'S <TRACK-NUMBER> (RETURN)'   - SEEK TRACK.
;
SHANDLER:
        JP      Z,SUBSM             ; IF 'M', SUBSM
;
;
;  SUBCOMMAND  ---'SEEK TRACK'---
;
SEEKR:
        BIT     DISKMODE,C
        JP      Z,ERRMODE
        CALL    L1NCR               ;E = TRACK #
SEEKNXT:
        LD      A,76                ;MAX TRACK #, MAXI DISK
        LD      D,39                ;MAX TRACK #, MINI DISK
        CALL    CHKNC               ;CHECK #
        JP      C,ERRTRAK
        LD      D,E                 ;TRACK #
        CALL    DSEEK
        LD      A,'S'               ;IN CASE OF SEEK ERROR
;
DERCKV:
       JP       DERRCK              ;DISK ERROR CHECK
;
;
;
;  SUBCOMMAND  ---'SUBSTITUTE MEMORY'---
;
;
SUBSM:
        SUB     A
        CALL    L1NCR
        EX      DE,HL               ; HL GETS ADDR
SM1:
        CALL    Z,PCADDR
        CALL    Z,SPACE
;
; PRINT CURRENT VALUE, REQUEST NEW VALUE &
; PRINT IT IF GIVEN
;
        CALL    PHN                 ;PRINT (HL)
        CALL    PMSGFOLLOWING
        DB      '.'+80H             ;THE PROMPT
        CALL    GCHR
        CP      '.'+1               ;IF <= '.',
        CALL    C,PCHR              ;NO SUBSTITUTION.
        JR      C,SM2
        EX      DE,HL
        CALL    GNHL                ;GET NEW VALUE
```

```
        EX      DE,HL
        LD      (HL),E
SM2:
        CP      CR
        CALL    NZ,SPACE
;
        RET     Z               ;IF CR, DONE.
        INC     HL
        LD      A,7             ;PRINT ADDRESS IF IT
        AND     L               ;IS A MULTIPLE OF 8
        JR      SM1
```

```
;
;       * * * MONITOR COMMAND * * *
;
; FUNCTION: READ DISK.
;
; FORMAT:'RD <DESTINATION-ADDR> <DESTINATION-END> <SECTOR-NUMBER> (RTN)
;                                       OR
;       'RD <DESTINATION-ADDR> S <SWATH-WIDTH> <SECTOR-NUMBER> (RTN)
;
RHANDLER:
        CP      'D'+CASE
        JP      NZ,INVCMD       ;ERROR INVALID CMD.
;
; READ DISK
;
READDR:
        CALL    SECSETUP
RD2:
        PUSH    BC
        PUSH    HL
        PUSH    DE
        CALL    DREAD
        LD      A,'R'           ;IN CASE OF READ ERROR
        CALL    NEXTSC          ;NEXT SECTOR (POPS STK.)
        JR      RD2
```

```
;
;       * * * MONITOR COMMAND * * *
;
; FUNCTION: WRITE DISK.
;
; FORMAT: 'WD <SOURCE-ADDRESS> <SOURCE-END> <SECTOR-NUMBER> (RETURN)'
;                                  OR
;         'WD <SOURCE-ADDR> S <SWATH-WIDTH> <SECTOR-NUMBER> (RETURN)'
;
WHANDLER:
        CP      'D'+CASE
        JP      NZ,INVCMD       ;ERROR INVALID CMD
;
; WRITE DISK
;
WRITDR:
        CALL    SECSETUP
WD2:
        PUSH    BC
        PUSH    HL
        PUSH    DE
        CALL    DWRITE
        LD      A,'W'           ;IN CASE OF WRITE ERROR
        CALL    NEXTSC          ;(POPS STACK)
        JR      WD2
;
;
; GET MEMORY ADDRESS, SECTOR # AND CHECK IT,
; AND LOAD B & C.
;
SECSETUP:
        BIT     DISKMODE,C
        JP      Z,ERRMODE       ;ERROR INVALID DISK MODE
        PUSH    BC
        CALL    L3NCRO          ;BUFFER ADDRS & SEC #
        POP     BC
        CALL    CHKSECNO
        JP      C,ERRSEC        ;ERROR INVALID SECTOR
;
;
; PRINT TRACK & SECTOR #'S
;
PTRKSC:
        IN      A,DTRACK
        LD      D,A
        EX      DE,HL
        CALL    PSNHL           ;PRINT TRACK AND SEC
        EX      DE,HL
        LD      A,C             ;DISK FLAGS
        LD      D,10            ;# OF RETRIALS
        RET
;
;
CHKSECNO:
        LD      A,26            ;MAX SEC #, MAXI DISK
        LD      D,18            ;MAX SEC #, MINI DISK
```

```
;
CHKNO:
        BIT     MAXI,C
        JR      NZ,CN2
        LD      A,D
CN2:
        CP      E
        RET     C
        LD      A,C
        AND     NDRIVES-1
        LD      B,A             ;DISK #
        LD      A,C             ;DISK FLAGS
        RET
```

```
;
;       * * * MONITOR COMMAND * * *
;
; FUNCTION: OUTPUT TO I/C PORT.
;
; FORMAT: 'O <DATA-BYTE> <PORT NUMBER> (RETURN)'
;
OUTP:
        CALL    GNHL
        EX      DE,HL           ;E GETS DATA
        CALL    L1NCR           ;GET PORT NUMBER
;
        LD      C,E             ; TO C
        OUT     (C),L
        RET
```

ERROR Reporting

```
;
;
INVCMD:                                 ;UNDEFINED COMMAND DETECTED
        CALL    PMSGFCLLOWING
        DB      CR,':-UNDEFINED COMMAND','?'+80H
        JP      CLEANSTACK
;
;
ERRSEL:                                 ;DISK SELECTION INVALID.
        CALL    PMSGFCLLOWING
        DB      CR,':-INVALID DISK NUMBER','?'+80H
        JP      CLEANSTACK
;
;
ERRMODE:                                ;INVALID DISK MODE.
        CALL    PMSGFCLLOWING
        DB      CR,':-INVALID DISK MODE','?'+80H
        JP      CLEANSTACK
;
;
ERRTRAK:                                ;INVALID TRACK SELECTION
        CALL    PMSGFCLLOWING
        DB      CR,':-INVALID TRACK SELECTION','?'+80H
        JP      CLEANSTACK
;
;
CRERR:                                  ;INVALID CR CHARACTER
        CALL    PMSGFCLLOWING
        DB      CR,':-CARRAGE RETURN REQUIRED','.'+80H
        JP      CLEANSTACK
;
;
HEXERR:                                 ;INVALID HEX CHARACTER
        CALL    PMSGFCLLOWING
        DB      CR,':-FIRST CHARACTER MUST BE IN HEX','.'+80H
        JP      CLEANSTACK
;
;
ERRSEC:                                 ;INVALID SECTOR.
        CALL    PMSGFCLLOWING
        DB      CR,':-INVALID SECTOR SELECTION','.'+80H
        JP      CLEANSTACK
;
;
NOIMP:                                  ;MSG IF COMMAND IS NOT IMPLEMENTED.
        CALL    PMSGFCLLOWING
        DB      CR,':-COMMAND NOT IMPLEMENTED','.'+80H
        JP      CLEANSTACK
;
;
;               RDOS 1.0 COMMAND TABLE
;
CMNDTBL:    ; CMD LOCATION TAG              COMMAND FUNCTION
        DW      NOIMP           ; A -
        DW      BOOTEC          ; B - BOOT CDOS (Boot Disk A, TRACE 0, sec1,
        DW      NOIMP           ; C -
```

```
        DW      DSPM            ; D - DISPLAY MEMORY
        DW      EXMINPUT        ; E - EXAMINE INPUT PORT
        DW      NOIMP           ; F -
        DW      GO              ; G - GO (TRANSFER OF CONTROL)
        DW      NOIMP           ; H -
        DW      INITIER         ; I - INITIALIZE SYSTEM CONSOLE
        DW      NOIMP           ; J -
        DW      KICKSTK         ; K - KICK SYSTEM STACK
        DW      NOIMP           ; L -
        DW      MOVE            ; M - MOVE A BLOCK OF MEMORY
        DW      NOIMP           ; N -
        DW      OUTP            ; O - OUTPUT To I/O PORT
        DW      NOIMP           ; P -
        DW      NOIMP           ; Q -
        DW      RHANDLER        ; R - READ DISK
        DW      SHANDLER        ; S - SUBSTITUTE MEM; SEEK TRACK (2)
        DW      NOIMP           ; T -
        DW      NOIMP           ; U -
        DW      VERIF           ; V - VERIFY BLOCKS OF MEMORY
        DW      WHANDLER        ; W - WRITE DISK
        DW      NOIMP           ; X -
        DW      NOIMP           ; Y -
        DW      NOIMP           ; Z -
;
LASTBYTE: EQU $-1
;
        END
```

Hex code in INTEL format

```
:10C00000000000F33E00ED47ED4FED46DD218000DE
:10C01000217C00F9EBCD5CC13E00D303DE34E6406C
:10C02000CA7CC0C379C1CDE9C30D3A2D564552491A
:10C0300046592054484154205448452043444F53C6
:10C04000204449534B20495320D4F554E544544AD
:10C05000204F4E2044524956452041 2E0D3A2D5036
:10C06000052455353203C52455455524E3E2057485A
:10C07000454E20524541445 9AECDEAC23E40D330F0
:10C08000DB301F38FBF33E10218000F9F50600CDB0
:10C09000A5C0200CF1F506C01E01CDF2C0CA80003B
:10C0A000F1EE1018E3CD31C1D3341698E6103E7F7F
:10C0B000D3043E0F281A3F0D1816F5CD31C1D334E6
:10C0C000ED511698F1173814E6203E1F28023E1D48
:10C0D000D330DB341F30FEDB3047A2C93E6FD304C3
:10C0E0003E18CDD0C0DB04E64020FA3E7FD3049753
:10C0F00047C9CD1CC1C688169CD330DB341F38D746
:10C10000EDA2C2FBC018CFCD1CC1C6A816FCD33013
:10C11000DB341F38C2EDA3C210C118B6CD31C1F651
:10C1200080577BD332DB34E6207AD3343E04C89781
:10C13000C9E6104F0497371710FDB1F62001338080
:10C14000C9DB00E640C8DE01C9CD41C128FBE67F61
:10C15000C990C0A09088848201CDEAC22151C10E4D
:10C16000003E19D302EDA3CD49C1CD49C1FE0D3E1C
:10C17000920F0C9CD47C31821CDB9C30D204152C4
:10C18000434144494 14E202052444F532020312EF8
:10C19000302020414354956458D9712D5DDE1DDCD
:10C1A000F9CD55C221A4C1DDE5E34ECB6923C4AE70
```

```
:10C1B000C3CDB9C3BBCD7CC32005DD360000C9D6D5
:10C1C00041DA94C4FE1AD294C45F1600CD7CC3FE3B
:10C1D0003BCA61C2EB291194C5195E2356EBFE4D93
:10C1E000E97BFE04D2AFC443DDE5E1F6B077545DF0
:10C1F0001378C64112CDDAC2FE3B2010CBBE13121B
:10C20000CDDAC2FE3B2005CBA6131297CDEAC2CBF6
:10C21000FE1ACDA5C03E48C8CDB9C30D3A2D524433
:10C220004F5320444554454354454420455252 4FB2
:10C23000522E0D3A2D455252 4F522053544154 55CF
:10C24000532043 4F4445202D2D2D3EA0CDB4C27820
:10C25000CD96C318273E0D185BCD47C34BED78181C
:10C26000EFCDB9C30D3A2D4449534B2053454C45AE
:10C270004354494F4E204552524F52BFC39FC1D9DC
:10C28000E1D9D12805152E90180A017FFFFD09FD85
:10C2900023E3160AE1C179D9E5D9CCD46CC430DCF6
:10C2A0001CCD78C4D0DB313C5FC5CDF1C3C1791E54
:10C2B00001C93E20F5CD41C1E67FFE1B28BEFE7DB3
:10C2C00028BADB00E68028FAF1F5E67FD301FE0DFF
:10C2D0002006CDB9C30A0080F1C9CD49C1CDB4C291
:10C2E000FE61D8D620C997CD26C3CD7DC3C208C56F
:10C2F000EBC9CD55C2CD91C33E3A18B8CD44C31A4F
:10C30000EDA12BC491C3C488C3EBC488C3C48EC33E
:10C31000C455C2EB2313EC18E6CD44C3E5D5C5ED03
:10C32000B0C1D1E118D9CD4DC3EBCD7DC3FE5320B3
:10C3300005CD4CC31807CD4DC3B7ED5223444DE591
:10C34000FDE1C997CD26C3CD4DC3189E97C52100E9
:10C3500000CD7DC3CD65C3DA29C5CDDAC2CD65C3B5
:10C360007830F7C1C947D630D8C6E9D8D6FA3003F5
```

```
:10C37000C607D8C60A2929292929B56FC997B7CCDAC3
:10C38000C2FE2028F9FE0FC9CDB2C27E1808CDB27A
:10C39000C27CCD96C37DCD9AC31F1F1F1F1FF5E61C
:10C3A0000FFE0A3802C607C630CDB4C2F1C9F57E09
:10C3B00023CDB4C21730F8F1C9E3CDAEC3E3C9E170
:10C3C000CD47C3EBE9C294C4CDE6C21610CDF2C28C
:10C3D000CD88C3EDA1E255C21528F07AE603CCB2B0
:10C3E000C218EDC394C4CA04C4CB69CACCC4CD4737
:10C3F000C33E4C1627CD7CC4DAE7C453CDBAC03E49
:10C4000053C317C297CD47C3EBCCF2C2CCB2C2CD57
:10C410008BC3CDB9C3AECDDAC2FE2FDCB4C23806B1
:10C42000EBCD4DC3EB73FF0DC4B2C2C8233E07A5CE
:10C4300018D7FE44C294C4CD5CC4C5E5D5CDF2C0C6
:10C440003E52CD7FC218F3FE44C294C4CD5CC4C535
:10C45000E5D5CD07C13E57CD7FC218F3CB69CACC15
:10C46000C4C5CD43C3C1CD78C4DA51C5DB3157EB68
:10C47000CD8EC3EB79160AC93E1A1612CB61200184
:10C480007ABBD879E6034779C9CD4DC3EBCD47C315
:10C490004BED69C9CDB9C30D3A2D554E44454649BA
:10C4A0004E454420434F4D4D414E44BFC39FC1CDE7
:10C4B000B9C30D3A2D494E56414C49442044495385
:10C4C0004B204E554D424552BFC39FC1CDB9C30D00
:10C4D0003A2D494E56414C49442044495534B204D36
:10C4E0004F4445BFC39FC1CDB9C30D3A2D494E56E8
:10C4F000414C49442054524143434B2053454C454301
:10C5000054494F4EBFC39FC1CDB9C30D3A2D4341CE
:10C5100052524147452052455455524E20524551A2
:10C52000554952454AEC39FC1CDB9C30D3A2D46BE
```

```
:10C530004952535420434841524143544552204D9F
:10C540005553542042452C494E20484558AEC39F7C
:10C55000C1CDB9C30D3A2D494E56414C49442053E3
:10C560004543544F522053454C454354494F4EAEDA
:10C57000C39FC1CDB9C30D3A2D434F4D4D414E44DC
:10C58000204E4F5420494D504C454D454E54454446
:10C59000AEC39FC173C526C073C5C5C359C273C599
:10C5A000BFC373C559C173C574C173C519C373C5FE
:10C5B00089C473C573C532C4E6C373C573C5FCC2F1
:08C5C00047C473C573C573C5C0
:00000001FF
```